I (d) Given $x_1, x_2, \ldots, x_n$ is a set. Define $x_1 \cdot \cdots \cdot x_n$ in $S_n$: $x_1 + x_2 + \cdots + x_n = S$. For every subset $y$ indicated by $y_1, y_2, \ldots, y_n$, check whether $x_1 + x_2 + \cdots + x_n = S$. If one of the tests succeeds, respond "yes" else respond "no". In half, there are $2^{n-1}$ subsets. Hence the algo. terminates.

One way to generate the subsets of $\{1, 2, \ldots, n\}$ is to represent each subset by a binary no. with $n$ bits in which a 1 indicates that the corresponding element is present. For example, 1011 indicates the set $\{4, 2, 1\}$.

(h) Let $M_x \neq \emptyset \Rightarrow L(M) \neq \emptyset$?

Method 1: If $M_x$ have $S$ states. If $M_x$ accepts any string, it must accept a string of length $< S$. Let the shortest string, $x_i \geq S$.

Proof: If $M_x$ accepts $x_i \geq S$. Let the In the sequence of states visited, $S$ must exist a repetition. If we delete the substring corresponding to any such repetition, the shorter string must be accepted too - contradicting the minimality of $M_x$.

Then check whether $M_x$ accepts at least one of strings in $\{v^0, v^1, v^2, \ldots, v^S\}$, i.e. attack all the strings of length $< S$. If $M_x$ respond "yes", reject. If "no", accept.

Method 2: Compute all the states of $M$ that can be reached from its initial state by a process analogous to the spread of a contagion. Technically
It is known as breadth-first scan.

Start with the set \( \{q_0\} \).

Let any stage let the set be \( A \).

For every \( q \in A \), if there is a transition to state \( q' \), include \( q' \) in \( A \).

If this process increases the size of \( A \), repeat the process.

Else, the set \( A \) gives all the states that can be reached from \( \{q_0\} \). Since the FA has \( s \) states and since repetition increases the size of the set, the max. number of repetitions is \( s-1 \). Hence the process is finite.

Check whether the resulting set \( A \) contains at least one final state. If so, respond yes. If not, respond no and halt.

(i) Let \( M_1 \) and \( M_2 \) have \( s_1 \) and \( s_2 \) states, respectively.

Method 1: If \( M_1 \) and \( M_2 \) accepts some common strings,

then they must accept a string of length \( \leq s_1 s_2 \).

Prove it directly or reduce to a simple extension of the above proof.

Then check whether some string of length \( \leq s_1 s_2 \) is accepted by both \( M_1 \) and \( M_2 \).

Method 2: Construct an FA for the language

\( L(M_1) \cap L(M_2) \).

This involves...
This involves running $M_1$ and $M_2$ in parallel; i.e. keeping track states of the form $(q, q')$ in which $q 
eq q'$ are states of $M_1$ and $M_2$, respectively. At the end, check that both the states are final states.

On the intersection as for, apply the methods

1. **Problem (b)**

   (b) $M_1, M_2, L(M_1) = L(M_2)$?

   We show $BTHP \leq_m BTHP$.

   **Typical Instance**

   Given $M_i$, we want to construct $M_1$ and $M_2$ s.t. $M$ halts on $BT$ iff $L(M_1) = L(M_2)$.

   Construct $M_1$ s.t. $L(M_1) = \{a, b^*\}$. — Easy.

   $M_2$: On any string of $a$'s and $b$'s, $M_2$ simulates $M$ on $BT$. If $M$ halts, then $M_2$ accepts $x$.

   Thus if $M$ halts on $BT$ then $L(M_2) = \{a, b^*\}$.

   If $M$ doesn't halt on $BT$ then $L(M_2) = \phi$.

   Hence $M$ halts on $BT$ iff $L(M_1) = L(M_2)$.

   Note the transformation of $L(M)$ to $[M, M']$ is a computable function.
I.(e) CFGs $G_1, G_2$ in $L(G_1) \cap L(G_2) \neq \emptyset$

we show that PCP \leq_m this part

Typical instances $(x_1,y_1), \ldots, (x_m, y_m) \in G_1, G_2$

\begin{align*}
E & = (x_1, y_1), \ldots, (x_m, y_m)
\end{align*}

we want to specify $G_1 \& G_2$ s.t. E has a solution iff $L(G_1) \cap L(G_2) \neq \emptyset$

\begin{align*}
G_1 : S & \rightarrow \alpha, S \alpha, \ldots, \mu S \kappa \\
G_2 : S & \rightarrow \gamma, S \gamma, \ldots, \nu S \kappa
\end{align*}

If $x_1, \ldots, x_m = y_1, \ldots, y_m$

then $E \in G_1 : S \rightarrow \ast x_1, \ast x_m, \ast \gamma_1, \ldots, \ast \gamma_m, \ast \kappa_1, \ldots, \ast \kappa_m$

in $G_2 : S \rightarrow \ast y_1, \ast y_m, \ast \gamma_1, \ldots, \ast \gamma_m, \ast \kappa_1, \ldots, \ast \kappa_m$

Hence there is string in \(L(G_1) \cap L(G_2)\).

Let \(G_1\) that is a common string \(z\), then \(z\) is of the form \(u w, u \in \Sigma^*, w \in \Sigma^*\).

\begin{align*}
\forall w = \gamma_1 \cdots \gamma_m \text{ then } u = x_1, x_m \text{ since } z \in L(G_1)
\end{align*}

\begin{align*}
\forall U = \gamma_1 \cdots \gamma_m \text{ since } z \in L(G_2)
\end{align*}

Hence $x_1, \ldots, x_m = y_1, \ldots, y_m$

Note that the transformation from $E$ to $G_1 \& G_2$

is a computable function.
III. Will show $\text{BTHP} \leq_m \text{this problem}$

Typical input $[M]$.

Given $[M]$, we transform it $[M']$ as

$TM \; M$ halts on $BT$ if $TM \; M'$ computes $f(n) = n^2$.

$M'$: Given any $n$, on a separate part of the tape, $M'$ simulates $M$. If $M$ halts, then $M'$ erases the computation part of the tape, then computes $n^2$ and halts.

$M$ halts on $BT \Rightarrow M'$ computes $f(n) = n^2$.

$M$ doesn't halt on $BT \Rightarrow M'$ computes $f(n) = \text{undefined}$ for every $n$.

Hence our goal is achieved.

Note also that the transformation from $[M]$ to $[M']$ is a computable function.